

Activity 5

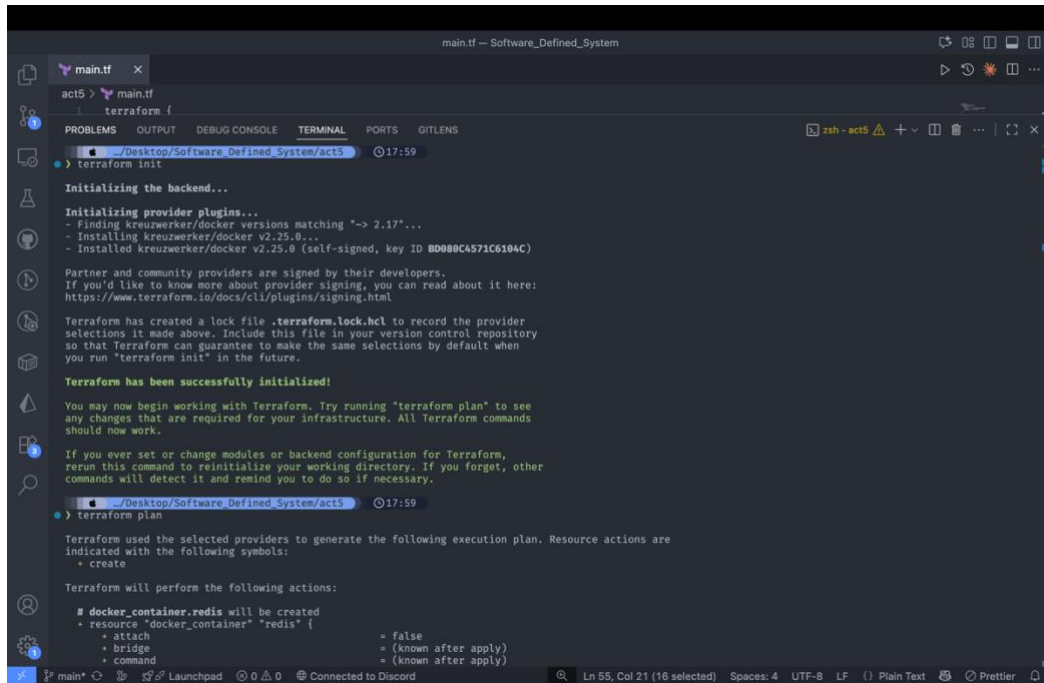
3. Prepare a PDF report containing the following information

a. Content of your terraform file.

```
terraform {  
  
  required_providers {  
    docker = {  
      source = "kreuzwerker/docker"  
      version = "~> 2.17"  
    }  
  }  
}  
  
provider "docker" {  
  host = "unix:///var/run/docker.sock"  
}  
  
resource "docker_network" "todo_net" {  
  name = "todo_network"  
}  
  
resource "docker_image" "redis_image" {  
  name      = "redis:latest"  
  keep_locally = false  
}  
  
resource "docker_container" "redis" {  
  name = "redis"  
  image = docker_image.redis_image.name  
  networks_advanced {  
    name = docker_network.todo_net.name  
  }  
  ports {  
    internal = 6379  
    external = 6379  
  }  
}  
  
resource "docker_image" "todo_image" {  
  name      = "natawut/todo-service:release-2.1"  
  keep_locally = false  
}  
  
resource "docker_container" "todo" {  
  name = "todo-service"
```

```
image = docker_image.todo_image.name
networks_advanced {
  name = docker_network.todo_net.name
}
ports {
  internal = 8000
  external = 8000
}
env = [
  "REDIS_HOST=redis",
  "REDIS_PORT=6379"
]
depends_on = [
  docker_container.redis
]
}
```

b. Use Screenshot of running terraform in each step



```
main.tf -- Software_Defined_System
main.tf x
act5 > main.tf
  terraform {
    • terraform init

Initializing the backend...

Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "~> 2.17"...
- Installing kreuzwerker/docker v2.25.0...
- Installed kreuzwerker/docker v2.25.0 (self-signed, key ID BD080C4571C6104C)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

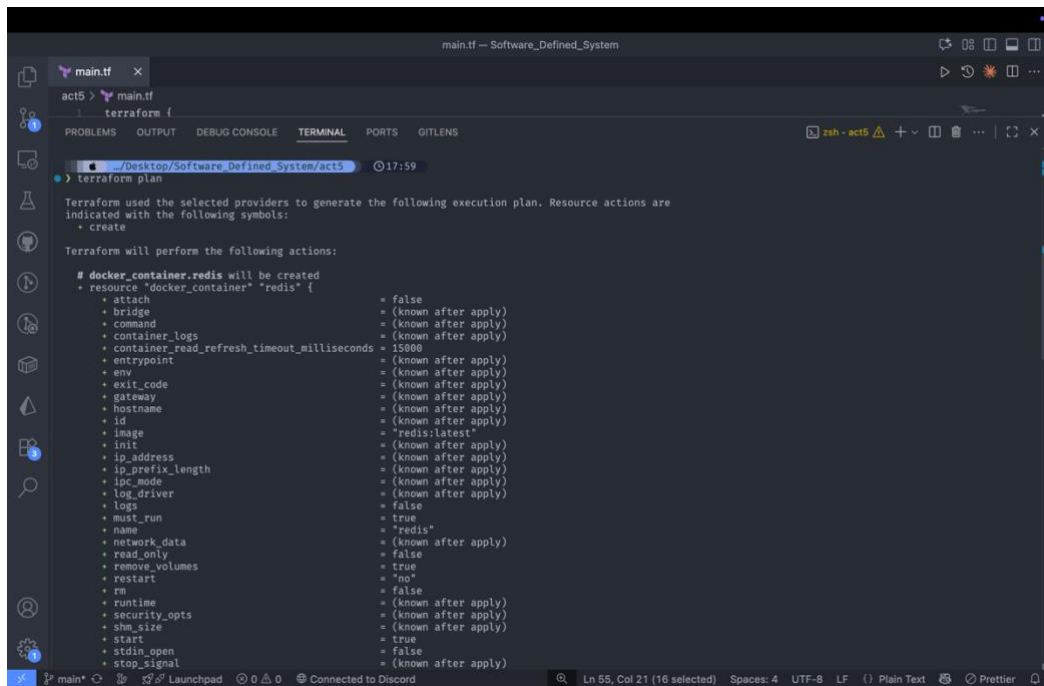
~/Desktop/Software_Defined_System/act5 17:59
• terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  • create

Terraform will perform the following actions:

# docker_container.redis will be created
+ resource "docker_container" "redis" {
  + attach      = false
  + bridge      = (known after apply)
  + command     = (known after apply)
  + container_logs = (known after apply)
  + container_read_refresh_timeout_milliseconds = 15000
  + entrypoint   = (known after apply)
  + env         = (known after apply)
  + exit_code    = (known after apply)
  + gateway      = (known after apply)
  + hostname     = (known after apply)
  + id          = (known after apply)
  + image        = "redis:latest"
  + init         = (known after apply)
  + ip_address   = (known after apply)
  + ip_prefix_length = (known after apply)
  + ipc_mode     = (known after apply)
  + log_driver   = (known after apply)
  + logs        = false
  + must_run     = true
  + name         = "redis"
  + network_data = (known after apply)
  + read_only    = false
  + remove_volumes = true
  + restart     = "no"
  + rm          = false
  + runtime      = (known after apply)
  + security_opts = (known after apply)
  + shm_size     = (known after apply)
  + start        = true
  + stdin_open   = false
  + stop_signal  = (known after apply)
```

terraform init



```
main.tf -- Software_Defined_System
main.tf x
act5 > main.tf
  terraform {
    • terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  • create

Terraform will perform the following actions:

# docker_container.redis will be created
+ resource "docker_container" "redis" {
  + attach      = false
  + bridge      = (known after apply)
  + command     = (known after apply)
  + container_logs = (known after apply)
  + container_read_refresh_timeout_milliseconds = 15000
  + entrypoint   = (known after apply)
  + env         = (known after apply)
  + exit_code    = (known after apply)
  + gateway      = (known after apply)
  + hostname     = (known after apply)
  + id          = (known after apply)
  + image        = "redis:latest"
  + init         = (known after apply)
  + ip_address   = (known after apply)
  + ip_prefix_length = (known after apply)
  + ipc_mode     = (known after apply)
  + log_driver   = (known after apply)
  + logs        = false
  + must_run     = true
  + name         = "redis"
  + network_data = (known after apply)
  + read_only    = false
  + remove_volumes = true
  + restart     = "no"
  + rm          = false
  + runtime      = (known after apply)
  + security_opts = (known after apply)
  + shm_size     = (known after apply)
  + start        = true
  + stdin_open   = false
  + stop_signal  = (known after apply)
```

terraform plan

terraform plan result

terraform apply

```

main.tf -- Software_Defined_System

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
~/Desktop/Software_Defined_System/acts 18:00
> terraform apply

# docker_image.todo_image will be created
+ resource "docker_image" "todo_image" {
+   id              = (known after apply)
+   image_id        = (known after apply)
+   keep_locally    = false
+   latest          = (known after apply)
+   name            = "natawut/todo-service:release-2.1"
+   output          = (known after apply)
+   repo_digest     = (known after apply)
+ }

# docker_network.todo_net will be created
+ resource "docker_network" "todo_net" {
+   driver          = (known after apply)
+   id              = (known after apply)
+   internal        = (known after apply)
+   ipam_driver     = "default"
+   name            = "todo_network"
+   options         = (known after apply)
+   scope           = (known after apply)
+ }

Plan: 5 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

docker_network.todo_net: Creating...
docker_image.todo_image: Creating...
docker_image.redis_image: Creating...
docker_image.todo_image: Creation complete after 0s [id=sha256:8ecf966b445201829f4459681b29e8b9dbae59b3caeb438411e27ed9e9ad82dnatawut/todo-service:release-2.1]
docker_image.redis_image: Creation complete after 2s [id=82d38fee475dd607343afc5e21de9549f22d5260cc5feae0c1e3bf4963ff8446]
docker_container.redis: Creating...
docker_container.redis: Creation complete after 1s [id=49e0b1a87d1a956ddd88b9d6ecde77508fa1274004ce1b37351dc501bc434f9]
docker_container.todo: Creating...
docker_container.todo: Creation complete after 0s [id=5dc8a8cb046f6b81827ed49191749a26c29b6f8273018f08fcfa50be1c8a519b]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

```

terraform apply result

```

main.tf -- Software_Defined_System

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
~/Desktop/Software_Defined_System/acts 18:00
> terraform apply

+ internal          = (known after apply)
+ ipam_driver       = "default"
+ name              = "todo_network"
+ options           = (known after apply)
+ scope             = (known after apply)
+ }

Plan: 5 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

docker_network.todo_net: Creating...
docker_image.todo_image: Creating...
docker_image.redis_image: Creating...
docker_image.todo_image: Creation complete after 0s [id=sha256:8ecf966b445201829f4459681b29e8b9dbae59b3caeb438411e27ed9e9ad82dnatawut/todo-service:release-2.1]
docker_image.redis_image: Creation complete after 2s [id=82d38fee475dd607343afc5e21de9549f22d5260cc5feae0c1e3bf4963ff8446]
docker_container.redis: Creating...
docker_container.redis: Creation complete after 1s [id=49e0b1a87d1a956ddd88b9d6ecde77508fa1274004ce1b37351dc501bc434f9]
docker_container.todo: Creating...
docker_container.todo: Creation complete after 0s [id=5dc8a8cb046f6b81827ed49191749a26c29b6f8273018f08fcfa50be1c8a519b]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

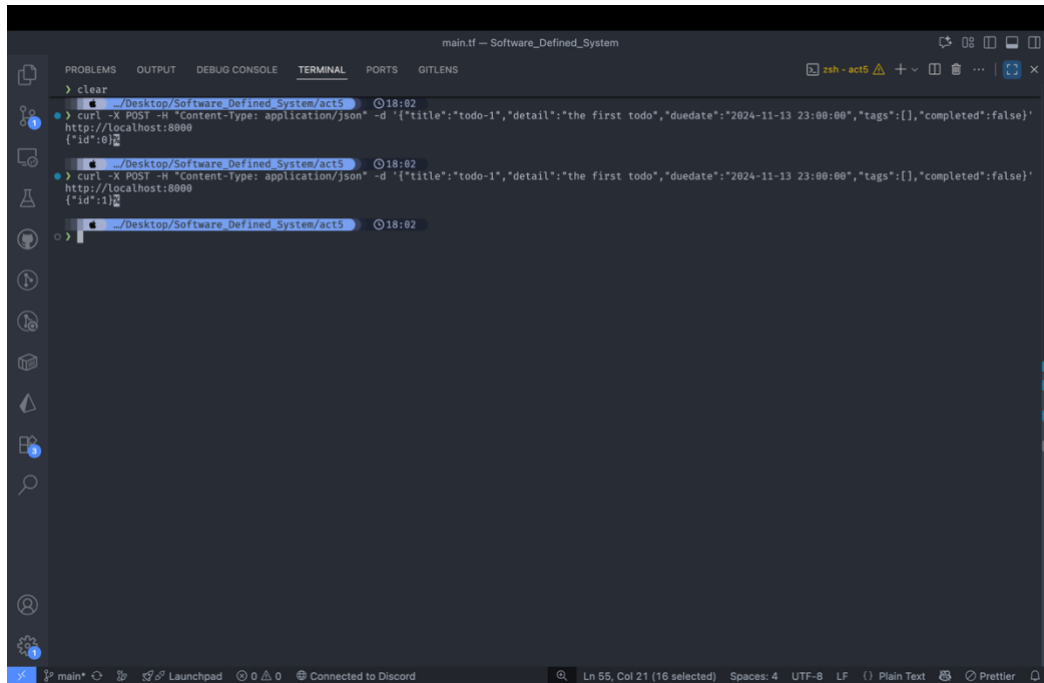
```

```

~/Desktop/Software_Defined_System/acts 18:01
> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                                NAMES
5dc8a8cb046f  natawut/todo-service:release-2.1    "/bin/sh -c 'python _"  12 seconds ago Up 11 seconds 0.0.0.0:8080->8080/tcp  todo-service
49e0b1a87d1a  redis:latest                        "docker-entrypoint.s..."  12 seconds ago Up 11 seconds 0.0.0.0:6379->6379/tcp  redis
ead018a32839  20f02e74d3ea                        "/nginx-ingress -ngi..." About a minute ago Up About a minute      k8s_nginx-ingress_ng
1nw-ingress-controller-5c0855a7fb-fp9xg_default_e2fc3b82-0132-4a2e-8f6f-91a10f0c3e95  "docker-entrypoint.s..."  2 minutes ago Up 2 minutes          k8s_redis_todo-servi
ce-redis-deployment-576ff46fd-6r74h_default_8ebd72f8-95c2-44a9-bea8-279d4f077d15_3  "nginx -g 'daemon of..."  2 minutes ago Up 2 minutes          k8s_nginx_nginx-depl
oyment-77d846869-9b72s_default_eb7ba98d-0a05-4fd6-919c-e198cdebfcf4_3  "/bin/sh -c 'python _"  2 minutes ago Up 2 minutes          k8s_todo-service_tod
o-service-redis-deployment-576ff46fd-6r74h_default_8ebd72f8-95c2-44a9-bea8-279d4f077d15_3  "nginx -g 'daemon of..."  2 minutes ago Up 2 minutes          k8s_nginx_nginx-depl
198d649be18f  7bbc8783b8ec                        "nginx -g 'daemon of..."  2 minutes ago Up 2 minutes          k8s_nginx_nginx-depl
oyment-77d846869-qbt7p_default_4131d9be-6264-4bc2-bad7-e1c627e65968_3

```

docker ps



The screenshot shows a VS Code terminal window with the title "main.tf - Software_Defined_System". The terminal is running a zsh shell. The commands and their outputs are as follows:

```
> clear

~/Desktop/Software_Defined_System/acts 18:02
> curl -X POST -H "Content-type: application/json" -d '{"title":"todo-1","detail":"the first todo","duedate":"2024-11-13 23:00:00","tags":[],"completed":false}' http://localhost:8000 {"id":0}

~/Desktop/Software_Defined_System/acts 18:02
> curl -X POST -H "Content-type: application/json" -d '{"title":"todo-1","detail":"the first todo","duedate":"2024-11-13 23:00:00","tags":[],"completed":false}' http://localhost:8000 {"id":1}
```

The status bar at the bottom indicates the file is "main.tf", the cursor is at "Ln 55, Col 21 (16 selected)", and the encoding is "UTF-8".

curl result